INFORMATION AND CONTROL 6, 137–146 (1963)

Discontinuity and Asymmetry in Phrase Structure Grammars*

G. H. MATTHEWS

Massachusetts Institute of Technology, Cambridge, Massachusetts, and MITRE Corporation, Bedford, Massachusetts

Chomsky (1959)[1] defines a class of grammars each of which contains a finite number of rules of the form $\chi_1 A \chi_2 \rightarrow \chi_1 \omega \chi_2$, where A is a single symbol and $\omega \neq I$. Such grammars are called context-sensitive phrase structure (CS) grammars. Context-free phrase structure (CF) grammars are a subclass of CS grammars all of whose rules are of the form $A \rightarrow \omega$, i.e., χ_1 and χ_2 are always null. A sequence of strings $(\varphi_1, \varphi_2, \cdots, \varphi_n)$ is called a φ-derivation of a grammar G if $\varphi = \varphi_1$ and for each i $(1 \leqq i < n)$, there are strings A, χ_1, χ_2, ψ_1, ψ_2 such that $\varphi_i = \psi_1 \chi_1 A \chi_2 \psi_2$, $\varphi_{i+1} = \psi_1 \chi_1 \omega \chi_2 \psi_2$, and $\chi_1 A \chi_2 \rightarrow \chi_1 \omega \chi_2$ is a rule of G. The language L_G is the set of strings which do not contain nonterminal symbols and which conclude S-derivations of the grammar G. Such a string is called a sentence of L_G.

DEFINITION 1. (i) A left-to-right (L-R) φ-derivation is a φ-derivation in which ψ_1 and χ_1 in the preceding paragraph are always strings of terminal symbols, i.e., $\varphi_i = yxA\chi\psi$, $\varphi_{i+1} = yx\omega\chi\psi$, and $xA\chi \rightarrow x\omega\chi$ is a rule of the grammar.

(ii) A right-to-left (R-L) φ-derivation is a φ-derivation in which χ_2

* This work was supported in part by the U. S. Army Signal Corps, the Air Force Office of Scientific Research, the Office of Naval Research, the National Science Foundation, and in part by the Air Force Electronic System Division, Air Force Systems Command, under Contract No. AF19(628)-2390 with The MITRE Corporation, and has been designated AFSC Technical Documentary Report No. AFESD-TDR-63-117. Further reproduction is authorized to satisfy the needs of the U. S. Government.

[1] The notations and terminology of that paper are used here: We use upper-case Latin letters for nonterminal strings; lower-case Latin letters for terminal strings; Greek letters for arbitrary strings; early letters of all alphabets for single symbols; later letters for strings of symbols; I for the null string; and S for the initial symbol. In addition, where χ is the string $\alpha_1\alpha_2\cdots\alpha_n$, χ^* is the string $\alpha_n\alpha_{n-1}\cdots\alpha_1$.

and ψ_2 in the preceding paragraph are always strings of terminal symbols, i.e., $\varphi_i = \psi\chi Axy$, $\varphi_{i+1} = \psi\chi\omega xy$, and $\chi Ax \rightarrow \chi\omega x$ is a rule.

DEFINITION 2. (i) A L-R grammar is one whose rule applications are restricted such that all of its derivations are L-R derivations.

(ii) A R-L grammar is one whose rule applications are restricted in such a way that all of its derivations are R-L derivations.

(iii) A one-way grammar is either a L-R grammar or a R-L grammar.

At any step in a derivation of a L-R grammar, only the leftmost nonterminal symbol in the string can be rewritten; the string of symbols to its left contains only terminal symbols. Similarly, in a derivation of a R-L grammar, strings of the type ψAy are rewritten $\psi\omega y$. The rules of a one-way CF grammar are of the same form as those of a CF grammar, and restricting the derivations of a CF grammar to one-way derivations obviously does not change the language generated by that grammar. However, it follows immediately from the definition of a one-way grammar that the left context in a rule of a L-R CS grammar, as well as the right context in a rule of a R-L CS grammar is restricted to strings of terminal symbols, i.e., the rules of a L-R grammar are CS rules of the form $xA\chi \rightarrow x\omega\chi$, and the rules of a R-L grammar are CS rules of the form $\chi Ax \rightarrow \chi\omega x$.

THEOREM 1. *For each one-way CS grammar there is an equivalent CF grammar.*

PROOF: We first consider the case of a L-R CS grammar. We can think

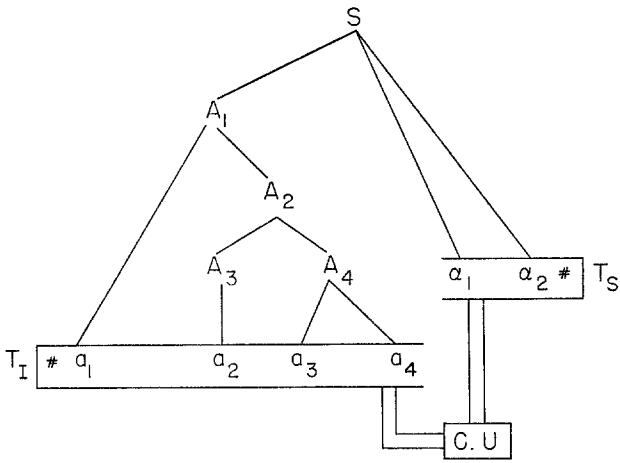

FIG. 1

of the derivations of the grammar as being carried out by a machine (see Fig. 1). The machine has a control unit that can interpret the rules of the grammar, and it has two tapes T_I and T_S. The machine can read the contents of both tapes; and, on the basis of these contents, either move the leftmost symbol of T_S to the right end of T_I, or replace the leftmost symbol of T_S by a string of symbols. At the beginning of a derivation T_I contains the boundary symbol # and T_S contains the string $S\#$. If at some step in the derivation of a sentence T_I contains the string yx and T_S contains $A\chi\psi$, then if there is a rule in the grammar $xA\chi \rightarrow x\omega\chi$, the machine may replace the A by ω. If at some step T_I contains x and T_S contains $a\psi$, then the machine will move the a from T_S to the right end of T_I, i.e., T_I will become xa and T_S will become ψ. Finally, when the only symbol remaining on T_S is the boundary symbol #, the machine will move this to the right end of T_I and stop. The operation of this machine is such that when it stops, T_S is blank and T_I contains a sentence of the language with boundary symbols at either end, and if some string x is a sentence of the language then there is a sequence of machine operations which will end with $\#x\#$ on T_I when the machine stops.

We now give a more precise description of this machine: But first we adopt some notational conventions.

Convention 1. By the expression $\omega = \omega^{(n)}$ we mean "the string ω is n symbols in length, or it is initial and/or final in a string, i.e., follows and/or precedes #, and is not greater than n symbols in length."

Convention 2. Where ω is a string, we use $\bar{\omega}$ to represent a string such that either ω is an initial part of $\bar{\omega}$, or $\bar{\omega}$ is an initial part of ω; i.e., for some ψ_1, ψ_2 where ψ_1 and/or ψ_2 is null, $\omega\psi_1 = \bar{\omega}\psi_2$.

We now construct a machine similar to the one described above in that it consists of two tapes and a control unit, but different in that it does

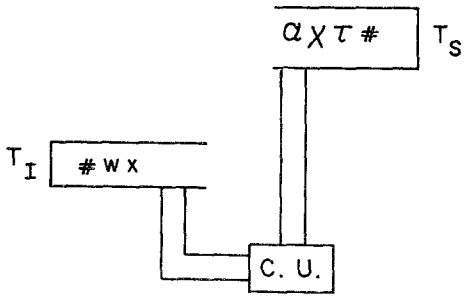

FIG. 2

not need to examine the contents of the tapes other than the leftmost symbol of T_S. The control unit consists of states which correspond to the set of environments which are significant for the particular grammar. These states are set up in the following manner. We first find a left context of a rule such that no rule has a longer left context; we call its length n. Similarly, we define m as the length of a right context of a rule such that no rule has a longer one. We now associate a state with each possible string $x\chi$ such that $x\chi = x^{(n)}\chi^{(m)}$. Let $S_{x,\chi}$ be such a state (see Fig. 2). Now consider a typical rule of the grammar $yA\psi \rightarrow y\alpha\omega\psi$. For each such rule there is a set of instructions, viz., all possible instructions of the form

$$(I,\ S_{zy,\psi\varphi}\ ,\ A) \rightarrow (S_{zy,\bar{\omega}}\ ,\ \alpha\omega) \tag{1}$$

This instruction can be interpreted as: The control unit writes the identity symbol on the right end of T_I when it both is in state $S_{zy,\psi\varphi}$ and is scanning the nonterminal symbol A at the left end of T_S; the control

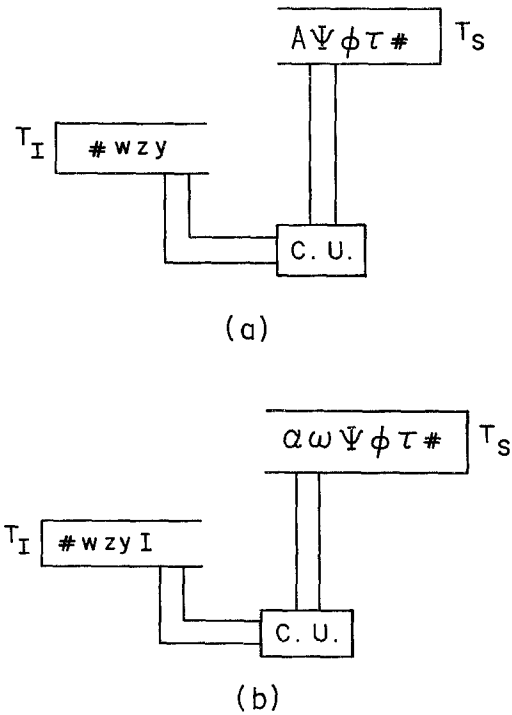

FIG. 3

unit then switches to state $S_{zy,\bar{\omega}}$ and replaces the symbol A by the string $\alpha\omega$ (see Fig. 3).

In addition to this set of instructions for each rule in the grammar, there is another set for each terminal symbol, the purpose of which is to move this symbol when it is the leftmost symbol of T_S from T_S to the right end of T_I. For each terminal symbol a, there are all possible instructions of the form

$$(a,\ S_{by,a\psi}\ ,\ a) \rightarrow (S_{ya,\psi\beta}\ ,\ \sigma) \tag{2}$$

When the control unit is in state $S_{by,a\psi}$ and is scanning the terminal symbol a on the left end of T_S, it writes a on the right end of T_I, switches to state $S_{ya,\psi\beta}$, and erases the leftmost symbol from T_S, i.e., the a it has just scanned. σ indicates that the symbol being scanned on T_S is erased. (See Fig. 4.)

We now have a description of a machine which generates the same language as does a L-R CS grammar. We will now reduce the instructions that describe this machine to those which describe a pushdown storage

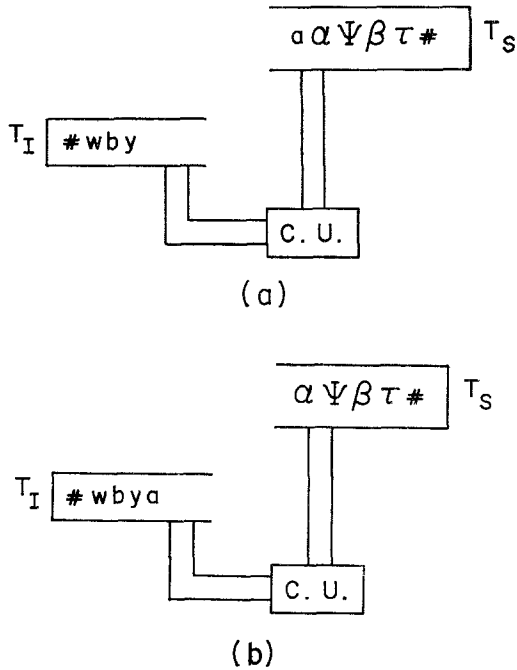

FIG. 4

automaton (PDS) (Chomsky, 1963). Instructions of type (2) are already in the form of a PDS instruction, but those of type (1) are not. Instead of writing the identity symbol on T_I, a PDS would just not read the next symbol on that tape. Not reading is indicated by the symbol e. Secondly a PDS instruction may either erase a single symbol from the left end of T_S, or it may write a string of symbols on the left end of T_S, but it cannot replace a symbol by a string as does (1). Replacement, however, can be simulated by two PDS instructions: The first erases the leftmost symbol of T_S and switches to an entirely new state, and the second writes the string on the left end of T_S.

$$(e,\ S_{zy,\psi\varphi}\ ,\ A) \rightarrow (S^{A}_{zy,\psi\varphi}\ ,\ \sigma) \tag{3}$$

$$(e,\ S^{A}_{zy,\psi\varphi}\ ,\ e) \rightarrow (S_{zy,\bar{\omega}}\ ,\ \alpha\omega) \tag{4}$$

These instructions are of the form that appear in a PDS. We now have a set of PDS instructions, i.e., (2), (3), (4), which accepts the language which is generated by instructions (1) and (2). However, these instructions assume that the PDS begins its computation with symbol S on the storage tape T_S, whereas a PDS computation actually begins with the single symbol σ on T_S. So we add a set of initial instructions which write S on the storage tape and switch the control unit from the initial state to each of the other states of the PDS

$$(e,\ S_0\ ,\ \sigma) \rightarrow (S_{x,\chi}\ ,\ S) \tag{5}$$

Finally, we add a set of final instructions which switch the control unit from each state to the initial state erasing σ from T_S.

$$(e,\ S_{x,\chi}\ ,\ \sigma) \rightarrow (S_0\ ,\ \sigma) \tag{6}$$

We can treat the case of a R-L CS grammar by reinterpreting instructions (1) and (2). This is because a R-L grammar is simply the mirror image of a L-R grammar. We turn the tapes T_S and T_I around, so that the control unit will either move the rightmost symbol of T_S to the left end of T_I when that symbol is terminal, or it will replace the rightmost symbol of T_S by a string when that symbol is nonterminal. The control unit is now in state $S_{x,\chi}$ when the string $x^*\omega$ is on T_I, and the string $\tau\chi^*$ is to the left of the rightmost symbol of T_S (see Fig. 5). Instruction (1) is now interpreted as: The control unit writes the identity symbol on the left end of T_I when it both is in state $S_{zy,\psi\varphi}$ and is reading the nonterminal symbol A at the right end of T_S; the control unit then switches

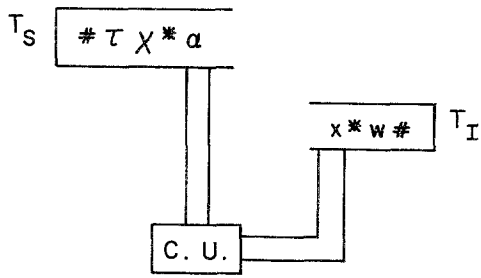

FIG. 5

to state $S_{zy,\bar{\omega}}$ and replaces the symbol A by the string $\omega^*\alpha$. This set of instructions corresponds to the rule $\psi^*Ay^* \rightarrow \psi^*\omega^*\alpha y^*$. Similarly, instruction (2) is interpreted as: When the control unit is in state $S_{by,\alpha\psi}$ and is reading the terminal symbol a on the right end of T_S, it writes a on the left end of T_I, switches to state $S_{ya,\psi\beta}$, and erases the rightmost symbol from T_S. With the instructions interpreted in this way, the machine will accept the string x^* if and only if the given R-L CS grammar generates the string x. Chomsky (1963) has proven that for each PDS there is an equivalent CF grammar; and Bar-Hillel, Perles, and Shamir (1960) have shown that if a set of strings x is a CF language, then so also is the set x^*. Q.E.D.

COROLLARY 1. *For each CF grammar—and thus for each PDS—there is an equivalent nondeterministic PDS containing just three types of instructions in addition to a single initial and a single final instruction. For each terminal symbol a of the grammar the PDS has the instruction* $(a, S_1, a) \rightarrow (S_1, \sigma)$; *for each nonterminal symbol A the PDS has the instruction* $(e, S_1, A) \rightarrow (S^A, \sigma)$ *and for each rule of the grammar* $A \rightarrow \omega$ *the PDS has the instruction* $(e, S^A, e) \rightarrow (S_1, \omega)$. *The initial and final instructions are* $(e, S_0, \sigma) \rightarrow (S_1, S)$ *and* $(e, S_1, \sigma) \rightarrow (S_0, \sigma)$, *respectively.*

PROOF: In the proof of Theorem 1 we set up a state to correspond to each possible significant context that might appear during the derivation of some sentence of the language. Then we constructed the set of all possible instructions of the form of (2). When we do this for a CF grammar, where there are no significant contexts, we need only the single state S_1. We then have all the instructions of the form

$$(a, S_1, a) \rightarrow (S_1, \sigma) \tag{7}$$

In the proof of Theorem 1, we set up a state for each combination of a nonterminal symbol and a possibly significant environment in which

that symbol may be rewritten by a rule. Then we constructed all possible pairs of instructions of the form of (3) and (4), where $yA\psi \rightarrow y\alpha\omega\psi$ is a rule of the grammar. If we apply this same construction to a CF grammar we get all possible pairs of instructions of the form

$$(e, S_1, A) \rightarrow (S^A, \sigma) \tag{8}$$

$$(e, S^A, e) \rightarrow (S_1, \omega) \tag{9}$$

where $A \rightarrow \omega$ is a rule. Applying the same reasoning with respect to the initial and final instructions we get for a CF grammar

$$(e, S_0, \sigma) \rightarrow (S_1, S) \tag{10}$$

$$(e, S_1, \sigma) \rightarrow (S_0, \sigma) \qquad \text{Q.E.D.} \tag{11}$$

DEFINITION 3. (i) A right discontinuous (RD) rule is one of the form

$$\chi_1 A \chi_2 \rightarrow \chi_1 \overset{n_1}{\underline{\ }} \alpha_1 \overset{n_2}{\underline{\ }} \alpha_2 \cdots \alpha_{m-1} \overset{n_m}{\underline{\ }} \alpha_m \chi_2$$

where $m \geqq 1$ and $n_i \geqq 0$ $(1 \leqq i \leqq m)$. A rule of this type rewrites a string of the form $\psi_1\chi_1 A\omega\psi_2$ as $\psi_1\chi_1\omega_1\alpha_1\omega_2\alpha_2 \cdots \alpha_{m-1}\omega_m\alpha_m\psi_2$ where $\omega_i = \omega_i^{(n)} i$ $(1 \leqq i \leqq m)$, and for some τ, $\omega\psi_2 = \omega_1\omega_2 \cdots \omega_m\psi_2 = \chi_2\tau$.

(ii) A left discontinuous (LD) rule is one of the form $\chi_1 A \chi_2 \rightarrow \chi_1\alpha_m \overset{n_m}{\underline{\ }} \alpha_{m-1} \overset{n_{m-1}}{\underline{\ }} \cdots \overset{n_2}{\underline{\ }} \alpha_1 \overset{n_1}{\underline{\ }} \chi_2$ where $m \geqq 1$ and $n_i \geqq 0$ $(1 \leqq i \leqq m)$. A LD rule rewrites a string of the form $\psi_1\omega A\chi_2\psi_2$ as $\psi_1\alpha_m\omega_m\alpha_{m-1} \cdots \omega_2\alpha_2\omega_1\chi_2\psi_2$ where $\omega_i = \omega_i^{(n)} i$ $(1 \leqq i \leqq m)$, and for some τ, $\psi_1\omega = \psi_1\omega_m\omega_{m-1} \cdots \omega_1 = \tau\chi_1$.

A discontinuous rule rewrites a nonterminal symbol as several symbols—at least one—and places each of these in the string at a certain distance—stated in the rule—from the rewritten symbol, to its right in the case of RD rules and to its left in the case of LD rules. For example, the RD rule $B_1AB_2 \rightarrow B_1\,\underline{3}\,C_1\,\underline{0}\,a\,\underline{1}\,C_2B_2$ rewrites the string $D_1B_1AB_2 \cdot D_2D_3D_4D_5$ as the string $D_1B_1B_2D_2D_3C_1aD_4C_2D_5$, i.e., the symbol A in the context preceded by B_1 and followed by B_2 is rewritten as the three symbols C_1, a, and C_2. These three symbols are then inserted in the string so that there are 3 symbols between B_1 and C_1, nothing between C_1 and a, and 1 symbol between a and C_2. If there are not enough symbols in the complete string, then identity elements are assumed (see Convention 1). The LD rule $B_1AB_2 \rightarrow B_1C_1\,\underline{2}\,a\,\underline{3}\,C_2\,\underline{0}\,B_2$ rewrites the string $D_1B_1AB_2D_2$ as $C_1aD_1B_1C_2B_2D_2$. Note that the rules defined in Chomsky (1959) are special cases both of RD rules and of LD rules, i.e., those in which $n_i = 0$ $(1 \leqq i \leqq m)$.

DEFINITION 4. (i) A RD grammar is a phrase structure grammar that contains just RD rules.

(ii) A LD grammar is one that contains just LD rules.

(iii) A discontinuous grammar is either a RD grammar or a LD grammar.

THEOREM 2. *For each discontinuous grammar there is an equivalent* CS *grammar.*

PROOF: This follows immediately from the fact that the effect of a discontinuous rule is to rewrite a string of symbols by a string which is not shorter, i.e., the RD rule given in Definition 3(i) can be replaced by a finite set of rules—all those of the form $\chi_1 A \omega \psi_2 \rightarrow \chi_1 \omega_1 \alpha_1 \omega_2 \alpha_2 \cdots \alpha_{m-1} \omega_m \alpha_m \psi_2$ where for some τ, $\omega\psi_2 = \chi_2\tau$ and τ and/or ψ_2 is null. Chomsky (1959) has shown that rules of this type can be replaced by a finite number of CS rules. Q.E.D.

DEFINITION 5. A one-way discontinuous grammar is either a L-R RD grammar or a R-L LD grammar.[2]

THEOREM 3. *For each one-way discontinuous grammar there is an equivalent* CF *grammar.*

PROOF: The proof of this theorem is similar to that of Theorem 1. Consider first the case of a L-R RD grammar. We construct a machine similar to the one we first constructed in the proof of Theorem 1. The control unit of this machine has a state for each possible sequence of symbols of the grammar x, χ where the length of x is the same as that of the longest left context that appears in the rules of the grammar, and the length of χ is the same as that of the longest string ω or χ_2 of Definition 3(i) that appears in the rules. Then for each RD rule we have the set of all possible instructions of the form

$$(I,\ S_{x\chi,\bar{\omega}}\ ,\ A) \rightarrow (S_{x\chi,\bar{\psi}}\ ,\ \omega_1\alpha_1\omega_2\alpha_2 \cdots \omega_m\alpha_m) \tag{12}$$

where $\omega = \omega_1\omega_2 \cdots \omega_m$ and where $\psi = \omega_1\alpha_1\omega_2\alpha_2 \cdots \omega_m\alpha_m$. When the control unit is in state $S_{x\chi,\bar{\omega}}$ and is scanning the nonterminal symbol A on the left end of T_S, it writes the identity symbol on the right end of T_I, switches to state $S_{x\chi,\bar{\psi}}$, and replaces the string $A\omega$, which is at the left end of T_S, by the string ψ. This machine also has all possible instructions of the form of (2). We can now replace instructions of type (12) by a finite sequence of PDS instructions similar to those of (3) and (4).

$$(e,\ S_{x\chi,\bar{\omega}}\ ,\ A) \rightarrow (S^{A1}_{x\chi,\bar{\omega}}\ ,\ \sigma) \tag{13}$$

[2] Yngve (1963) has investigated the application to natural language of a special class of L-R RD grammars—those in which for all rules $m \leqq 2$, $n_1 = 0$, and $n_2 \leqq 1$.

$$(e,\ S^{A1}_{zy,\bar{\omega}}\ ,\ e) \rightarrow (S^{A2}_{zy,\bar{\omega}}\ ,\ \sigma) \tag{14}$$

$$(e,\ S^{A2}_{zy,\bar{\omega}}\ ,\ e) \rightarrow$$

$$\vdots \tag{15}$$

$$\rightarrow (S^{A\,n_1+n_2+\cdots+n_m}_{zy,\bar{\omega}}\ ,\ \sigma) \tag{16}$$

$$(e,\ S^{A\,n_1+n_2+\cdots+n_m}_{zy,\bar{\omega}}\ ,\ e) \rightarrow (S^{A}_{zy,\bar{\omega}}\ ,\ \sigma) \tag{17}$$

$$(e,\ S^{A}_{zy,\bar{\omega}}\ ,\ e) \rightarrow (S_{zy,\bar{\psi}}\ ,\ \omega_1\alpha_1\omega_2\alpha_2\ \cdots\ \omega_m\alpha_m) \tag{18}$$

For the case of a R-L LD grammar, we can reinterpret instructions like (12) as we did in the proof of Theorem 1, so that they generate the string x^* just if x is a sentence generated by the given R-L LD grammar. Q.E.D.

COROLLARY 2. *Given a one-way discontinuous grammar, there is an algorithm for constructing a CF grammar equivalent thereto.*

PROOF: Both our proof of the equivalence of one-way discontinuous grammars to PDS and Chomsky's (1963) proof of the equivalence of PDS to CF grammars are constructive. Thus, they present the algorithm for constructing the required CF grammar.

Theorems 1 and 3 show that if the derivations of a grammar are confined to one-way derivations, the grammar contains the same inadequacies for the description of natural languages as do CF grammars. And these inadequacies are not overcome when such a grammar is supplemented by discontinuous rules. Theorems 2 and 3 show that discontinuous rules do not increase the generative power of one-way grammars or of CS grammars; but the effect of CF discontinuous rules on the generative power of CF grammars is still to be determined.

RECEIVED: December 18, 1962

REFERENCES

BAR-HILLEL, Y., PERLES, M., AND SHAMIR, E., (1960), On formal properties of simple phrase structure grammars. Tech. Rept. No. 4, Applied Logic Branch, Hebrew University of Jerusalem.

CHOMSKY, N., (1959), On certain formal properties of grammars. *Inform. and Control* **2**, 137–167.

CHOMSKY, N., (1963), Formal properties of grammars. *In* R. R. BUSH, E. H. GALANTER, R. D. LUCE (eds.) "Handbook of Mathematical Psychology," Vol. 2. Wiley, New York.

YNGVE, V. H., (1962), Random generation of English sentences. In *Proc. 1961 Intern. Congr. Machine Translation of Languages and Appl. Language Analysis, Teddington, England.* National Physical Laboratory, Teddington, England.